Gayathri Ravichandran

5437937197

Scala version : 2.11

Spark : 2.3.1

Task 1:

To run :

./spark-submit --class Task1 GayathriDM4.jar <path to input file> <feature ><N><iteration>

Explanation of the Algorithm:

For task1 , I implement the k-means clustering algorithm with 1) Word count and 2)TF-IDF Features using Euclidean distance measure.

I create TF and TF-IDF features for the input file using in-built functions. I create 2 hash maps - (a) document_cluster_hash to store document ids as keys and their corresponding clusters, and (b) cluster_centroid_hash to store the centroid of every cluster.

For every iteration, I find which document belongs to which cluster based on minimum Euclidean distance between the point and the centroid of the cluster. After this, I calculate the error per cluster using Vectors.sqdist(). To find the total WSSE, I sum the squares of errors for every cluster.

To find the top ten words, I first find all the words belonging to each cluster and find the frequency, and then order it by their frequency count.

All these values are then written to file.

****
****

Task 2:

To run :

./spark-submit --driver-memory 6g --class Task2 GayathriDM4.jar <path to input file> <algorithm ><N><iteration>

Explanation:

For task2, I implement the k-means clustering algorithm as well as Bisecting K-means algorithm.

For k-means,I create TF-IDF features for the input file using in-built functions. I train the Kmeans model based on these features, and use Kmeans.predict().zipWithIndex() to retrieve tuples of (document_id, cluster_id). This tells us which documents belong to which clusters.

For bisecting K-means,,I create TF-IDF features for the input file using in-built functions. I train the Bisecting Kmeans model based on these features, and use model.predict().zipWithIndex() to retrieve tuples of (document_id, cluster_id). This tells us which documents belong to which clusters.

For both these implementations:

I calculate the error per cluster using Vectors.sqdist(). To find the total WSSE, I sum the squares of errors for every cluster.

To find the top ten words, I first find all the words belonging to each cluster and find the frequency, and then order it by their frequency count.

All these values are then written to file.

****
****